Coreference detection in XML metadata

Marcin Szymczak*† Sławomir Zadrożny* Guy De Tré†
* Systems Research Institute
Polish Academy of Sciences, Warsaw, Poland
† Department of Telecommunications and Information Processing
Ghent University, Ghent, Belgium

***Abstract*—Preserving data quality is an important issue in data collection management. One of the crucial issues hereby is the detection of duplicate objects (called coreferent objects) which describe the same entity, but in different ways. In this paper we present a method for detecting coreferent objects in metadata, in particular in XML schemas. Our approach consists in comparing the paths from a root element to a given element in the schema. Each path precisely defines the context and location of a specific element in the schema. Path matching is based on the comparison of the different *steps* of which paths are composed. The uncertainty about the matching of steps is expressed with *possibilistic truth values* and aggregated using the Sugeno integral. The discovered coreference of paths can help for determining the coreference of different XML schemas.**

I. Introduction

The existence of duplicate data across multiple related databases significantly lowers data quality and should be avoided. With duplicate data, hereafter called *coreferent data*, we mean data pieces which refer to the same real world entity, but differ in some way, e.g., due to using a different modelling or representation. As an example, Figure 1 shows two data pieces resp. taken from FreeDB[1] (right tree) and Discogs[2] (left tree) which describe the same entity (concept) *'compact disc'*, but in different ways.

The detection and handling of coreferent data can be based only on data or on both data and metadata. In this paper we present a method for detecting coreferent data (schema level objects) in metadata. Such methods can then later on be used for improving the coreference detection of the data described by the metadata. *XML* [1] was chosen as the data model for multiple reasons. First, XML is one of the most popular formats to store and exchange data. Second, it consists of two well defined layers: a metadata layer (XML schema) and a data layer (XML document). Moreover, it is platform independent and often contains information that is represented in different ways.

Paths can be considered as one of the main components of an XML schema. Each path precisely defines the context and location of a specific element. The objective of this paper is to propose a method for detecting coreferent elements in XML schemas based only on metadata and also, as a next step, a method for detecting coreference of XML schemas. More specifically, the subproblem of coreference detection based on coreferent paths is studied in the paper.

[1]FreeDB, http://www.freedb.org/
[2]Discogs, http://www.discogs.com/data/

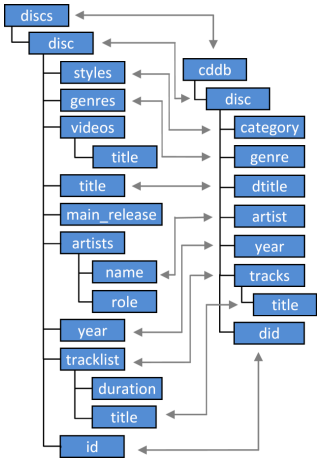

Fig. 1. Real-world data example: sample of schema instance from FreeDB (right tree) and from Discogs (left tree).

The rest of this paper is structured as follows. In Section II some preliminary notions are explained. Next, in Section III it is studied how coreferent XML elements can be detected based on *paths*. Section IV reports some experimental results. Finally, Section V summarises the contributions of this paper.

II. Preliminaries

Within the scope of this paper it is assumed that entities from the real world are described as *objects* which are characterized by a number of features (e.g., an object *person* may be characterized by a *name*, *address*, etc.). In case of XML, features include *elements* and *attributes*. Their properties and relationships are defined in an XML schema. Each XML schema comprises a collection M of metadata. Fundamental types of metadata included are element and attribute declarations and data types definitions. As elements may be nested in other elements some *paths* leading from a document root element to a leaf element are also included.

Considering a more abstract view of entity representation, denoting the universe of the i^{th} feature of an object by U_i, we can model the universe O of objects by:

$$O = U_1 \times \cdots \times U_n. \tag{1}$$

Two objects $o_1 \in O$ and $o_2 \in O$ are said to be *coreferent* (denoted $o_1 \leftrightarrow o_2$) if and only if they describe the same real world entity. Moreover, two objects $o_1 \in O$ and $o_2 \in O$ are said to be coreferent based on metadata (denoted $o_1 \overset{M}{\leftrightarrow} o_2$)

if and only if metadata M_1 of features of o_1 are coreferent to metadata M_2 of features of o_2, i.e., if o_1 and o_2 have a coreferent *structure*.

Consequently, $\overset{M}{\leftrightarrow}$ is an equivalence relation (reflexive, symmetrical and transitive). From a mathematical point of view this means that given a database D of objects, identification of coreferent objects in D comes down to defining such a partition $\mathbb{E}$ of D that objects within each partition class are coreferent and objects in different partition classes are not coreferent. Hence, this boils down to finding clusters in D.

Two elementary operators play an important role in establishing a partition of a set of objects: a *comparison operator* working at the level of object features (or metadata features) and an *aggregation operator* combining the comparison scores obtained for particular features.

Definition 1 (Comparison operator): A comparison operator on the universe O is defined by a function $\mathcal{C}$:

$$\mathcal{C} : O^2 \to \mathbb{L} \tag{2}$$

where $(\mathbb{L}, \leq)$ is a totally ordered and bounded lattice.
A comparison operator $\mathcal{C}$ compares (a feature of) two objects o_1 and o_2 and expresses the result of this comparison as a *matching degree* belonging to a totally ordered and bounded lattice $\mathbb{L}$. In the case of probabilistic methods, $\mathbb{L}$ can be instantiated with the unit interval $[0, 1]$ in order to express the result of comparison as a probability of identity of the objects. Other practical examples of $\mathbb{L}$ are the set of Boolean values $\mathbb{B} = \{T, F\}$, where T and F denote *true* and *false* or the set of possibilistic truth values (PTVs) $\mathcal{F}(\mathbb{B})$ [2].

In our approach we use PTVs to express the (uncertainty about the) results of a comparison. Hereby, a PTV is a (normalized) possibility distribution [3] defined over the set of Boolean values $\mathbb{B}$ [4]. A PTV expresses the uncertainty about the Boolean value of a *proposition* p. In the context considered here, the propositions p of interest are of the form

$$p = o_1 \text{ and } o_2 \text{ are coreferent}$$

where o_1 and o_2 are two given (metadata) objects.

Let P denote a set of all propositions under consideration, then each $p \in P$ can be associated with a PTV denoted $\tilde{p} = \{(T, \mu_{\tilde{p}}(T)), (F, \mu_{\tilde{p}}(F))\}$, where $\mu_{\tilde{p}}(T)$ represents the possibility that p is true and $\mu_{\tilde{p}}(F)$ denotes the possibility that p is false. The domain of all possibilistic truth values is denoted $\mathcal{F}(\mathbb{B})$, i.e., the fuzzy power set of (normalised) fuzzy sets over $\mathbb{B}$.

Let us define the order relation $\geq$ on the set $\mathcal{F}(\mathbb{B})$ by:

$$\tilde{p} \geq \tilde{q} \Longleftrightarrow \begin{cases} \mu_{\tilde{p}}(F) \leq \mu_{\tilde{q}}(F), & \mu_{\tilde{p}}(T) = \mu_{\tilde{q}}(T) = 1 \\ \mu_{\tilde{q}}(T) \leq \mu_{\tilde{p}}(T), & \text{else.} \end{cases} \tag{3}$$

Comparison of complex objects is usually a two-stage process. First, parts of objects, notably values of their (metadata) features, are compared using a comparison operator. Thus, we extend our definition of the comparison operator (2) so as to make it applicable also to scalar feature values:

$$\mathcal{C}_i : U_i^2 \to \mathbb{L} \tag{4}$$

In this way a separate comparison operator $\mathcal{C}_i$ can be defined for each feature. Then, the results of those comparisons are aggregated to obtain an overall PTV reflecting the coreference of the whole objects being compared. Therefore, another elementary operator, an *aggregation operator*, is needed.

Definition 2 (Aggregation operator): An aggregation operator on $\mathbb{L}$ is defined by a function $\mathcal{A}$:

$$\mathcal{A} : \mathbb{L}^n \to \mathbb{L} \tag{5}$$

where $(\mathbb{L}, \leq)$ is a totally ordered and bounded lattice.
For more information on aggregation operators the reader is referred to [5]. We assume that $\mathcal{A}$ is monotone in the following sense: $\forall \boldsymbol{l}, \boldsymbol{l}' \in \mathbb{L}^n : \boldsymbol{l} \leq \boldsymbol{l}' \Rightarrow \mathcal{A}(\boldsymbol{l}) \leq \mathcal{A}(\boldsymbol{l}')$ where the relation $\leq$ is generalized to vectors from $\mathbb{L}^n$ in a point wise way.

Based on the definition of these two elementary operators, pairwise comparison of two objects can be generally written as:

$$\mathcal{C}(o_1, o_2) = \underset{i=1}{\overset{n}{\mathcal{A}}} \left(\mathcal{C}_i \left(u_{i1}, u_{i2}\right)\right) \tag{6}$$

where u_{i1} and u_{i2} denote the value of the i^{th} (metadata) feature of o_1 and o_2, respectively.

Aggregation of PTVs may be carried out using the *Sugeno integral for possibilistic truth values* as defined in [6]; cf. also [7]. We will briefly remind its idea in what follows.

This integral uses two fuzzy measures (γ^T and γ^F). The measure γ^T (resp. γ^F) provides the conditional necessity that two complex objects are (not) coreferent, given that some set of (metadata) features are (not) coreferent. As required by the definition of fuzzy measures, γ^T and γ^F are monotonic and are normalised between $\emptyset$ and $\mathcal{L}$, where $\mathcal{L}$ represents the appropriate set of labels, which are used to name the features. The label of a feature represents the class of entities (values) that can be used to describe that feature.

Definition 3 (Sugeno integral for PTVs): Given a set of propositions $P = \{p_1, ..., p_n\}$ and a corresponding set of PTVs $\tilde{P} = \{\tilde{p}_1, ..., \tilde{p}_n\}$, let γ^T and γ^F be two fuzzy measures defined on *P* which satisfy the condition:

$$\forall Q \subseteq P : \min(\gamma^T(Q), \gamma^F(\bar{Q})) = 0 \tag{7}$$

Then the Sugeno integral of $\tilde{P}$ with respect to γ^T and γ^F is defined by:

$$S_{\gamma^{T,F}}(\tilde{P}) : \mathcal{F}(\mathbb{B})^n \to \mathcal{F}(\mathbb{B}) : \tilde{P} \mapsto \tilde{p}, \text{ where} \tag{8}$$

$$\mu_{\tilde{p}}(T) = 1 - \bigvee_{i=1}^{n} Nec\Big(\tilde{P}_{(i)^F} = F\Big) \wedge \gamma^F\Big(\tilde{P}_{(i)^F}\Big) \tag{9}$$

$$\mu_{\tilde{p}}(F) = 1 - \bigvee_{i=1}^{n} Nec\Big(\tilde{P}_{(i)^T} = T\Big) \wedge \gamma^T\Big(\tilde{P}_{(i)^T}\Big) \tag{10}$$

where $\cdot_{()^T}$ (respectively $\cdot_{()^F}$) is a permutation that orders the elements of $\tilde{P}$ according to the largest (smallest) PTV first.

III. XML PATHS IN COREFERENCE DETECTION

An XML schema contains various types of metadata which define the structure, data types, restrictions etc. that apply on the corresponding XML documents. These metadata allow it to reconstruct *paths* by which we mean here sequences of elements connecting a root element with leaf elements.

Moreover, each XML element can be defined within the scope of a *namespace*. XML namespaces comprise logically related sets of names. These names have to be unique within a namespace but may be freely repeated in distinct namespaces. XML namespaces will not be considered in this paper.

Two elements in two XML schemas may refer to the same feature of a real-world entity (may be *coreferent*) even if they have different tags and are located at different levels of an XML file. Comparing their paths, as defined above, may help to discover their coreference. In this section a novel method for detecting coreferent XML elements based only on comparing their paths is presented. However, the method is extendable to other metadata like data types and restrictions, e.g., using machine learning techniques [8], but this is out of the scope of this paper. The proposed method is purely syntactic and does not use any other external information sources like ontologies or dictionaries. Using such extra information can increase the quality of coreference detection, but also requires more computational resources and is not being considered in the paper. The following steps describe our object (XML element) coreference detection method based on paths.

Step 1: Extraction

First, the algorithm extracts elements (leaves) with metadata (paths to the root) from the input XML schemas to be compared, cf., both schemas in Figure 1. For example, a path */cddb/disc/tracks/title* is extracted for the element `title` of the schema on the right. Thus, two sets of paths are obtained. In our example, elements (more precisely, their paths) extracted from the schema on the left are represented as a set A (not a multiset as each path in XML schema is unique) and elements extracted from the schema on the right are represented as a set B.

Step 2: Generation of a coreferent paths matrix

A coreferent paths matrix P is generated based on the technique that has been introduced in [9]. For two input schemas with respectively m and n paths, P is a $m \times n$ matrix, where element $P_{i,j}$, $i = 1, \ldots, n$, $j = 1, \ldots, m$ is the PTV resulting from the comparison of paths i and j from both schemas and reflecting the (uncertainty about) their coreference. An example is given in Table I. Each row corresponds to a path from A, whereas each column corresponds to a path from B. The matrix elements are computed using the following substeps.

1) Tokenization: In this step, each path is tokenized what results in a list of substrings (elements on the path) which are called *steps* [2]. Thus, tokenization transforms a path into a list of steps. In most cases, steps are separate words. In our approach, tokenization of a path is equivalent to deleting all delimiting '/' characters in the paths. For instance tokenization of the path */cddb/disc/tracks/title* results in a list $[cddb, disc, tracks, title]$. The reason why we introduce tokenization is that traditional string comparison methods (character based methods) are not efficient for long character strings [2]. A major advantage of tokenization is that it decreases complexity and enhances the effectiveness of string comparison in our approach. Moreover, it allows us to introduce a specific aggregation method which helps to detect coreferent elements more effectively.

TABLE I
EXAMPLE OF A COREFERENT PATHS MATRIX.

Paths	/cddb/disc/dtitle	/cddb/disc/artist	/cddb/disc/year	/cddb/disc/tracks/title
/discs/disc/title	**T:1.00 F:0.12**	T:1.00 F:0.54	T:1.00 F:0.63	T:1.00 F:0.36
/discs/disc/artists/name	T:1.00 F:0.81	**T:1.00 F:0.44**	T:1.00 F:0.81	T:1.00 F:0.86
/discs/disc/year	T:1.00 F:0.63	T:1.00 F:0.63	**T:1.00 F:0.06**	T:1.00 F:0.81
/discs/disc/tracklist/title	T:1.00 F:0.36	T:1.00 F:0.54	T:1.00 F:0.81	**T:1.00 F:0.16**

2) Steps comparison: A one-level string comparison technique proposed in [2] is used to compare the resulting steps from each pair of paths. This low level comparison method estimates the possibility that two given steps are coreferent and is based on an approximation of *weak intersections*. It uses the concept of a moving window to construct the intersection of the two input steps. This technique has been chosen because of its efficiency [2]. In the literature a multitude of algorithms for string comparison has been proposed and these may also be employed here. An example of an interesting survey concerning strings in general is [10]. Work focused on coreference (duplicates) detection in the context of XML is [11]. An example of an approach employing fuzzy logic which might also be of interest to the reader is [12].

The algorithm compares pairs of steps from one input list with steps from the other input list. It generates PTVs which express the uncertainty about the coreference of the compared steps as described in [2]. The possibility that a proposition p, stating that two steps are coreferent, is true is calculated as the ratio between the number of common characters and the length of the longer step. On the other hand, the possibility that p is false is computed as the ratio between the number of different and missing characters, counted with specific weights, and the length of the longer step. Both possibilites are normalized (divided by the maximum of these possibilities). The PTVs resulting from the comparisons of all steps in the two paths are represented in a so-called *coreferent steps matrix*. An example of such a matrix is given in Table II.

Our step comparison method thus takes into account misspellings and abbreviations and moreover has a low computational complexity. This is a great advantage for XML coreference detection because abbreviations are frequently used in XML paths.

3) Mapping at the steps level: This substep selects the best 1:1 mapping between steps belonging to the lists representing two paths in the coreferent steps matrix (cf. Table II). The

TABLE II
EXAMPLE OF A COREFERENT STEPS MATRIX.

Steps	cddb	disc	tracks	title
discs	**T: 0.0** **F: 1.0**	T: 1.0 F: 0.01	T: 0.0 F: 1.0	T: 0.0 F: 1.0
disc	T: 0.0 F: 1.0	**T: 1.0** **F: 0.0**	T: 0.0 F: 1.0	T: 0.0 F: 1.0
tracklist	T: 0.0 F: 1.0	T: 0.0 F: 1.0	**T: 1.0** **F: 0.17**	T: 0.0 F: 1.0
title	T: 0.0 F: 1.0	T: 0.0 F: 1.0	T: 0.0 F: 1.0	**T: 1.0** **F: 0.0**

mapping algorithm [9] first selects the largest PTV for each row of the matrix (largest in the sense of the order relation (3)); if it is not unique one of the largest values is chosen randomly. Next, the algorithm checks if conflicts occur, i.e., if there is more than one PTV selected in the same column. If this is the case, then the conflicts are resolved in a way best illustrated with the coreferent steps matrix shown in Table II. For rows 'discs' and 'disc' a conflict occurs as both of them best match column 'disc'. However row 'disc' contains a larger PTV as $(1, 0) \geq (1, 0.01)$, so column 'cddb' is chosen as matching row 'discs' (column 'cddb' is the only remaining unselected). Finally, no conflicts occur and the algorithm stops. The selected PTVs, indicating matching pairs of rows and columns, are shown in bold, e.g., the steps *tracks* and *tracklist* are chosen as matching and the PTV expressing their matching degree is equal $(1.0, 0.17)$. Our example only illustrates a simple case of a conflict. For a detailed description of a conflict resolution procedure the reader is referred to [9].

4) Aggregation at the steps level: The last substep in the generation of the coreferent paths matrix is the aggregation of the PTVs in the coreferent steps matrix, hereby using the mappings that were obtained in the previous substep. The result of this aggregation is a PTV for each pair of paths which expresses the uncertainty about the coreference of these two paths. For the aggregation at the steps level, the following issue should be dealt with. Consider path */cddb/disc/tracks/title* from the right schema and path */discs/disc/tracklist/title* from the left schema of Figure 1, where not all steps are matched at the beginning of the paths. In such a case, both paths have a different context, but may describe the same object. This means that if the steps at the end of the paths are coreferent then this is a strong hint that the paths are coreferent. So, it should be clear that not all steps in a path are equally important with respect to coreference detection. Generally, two paths are more likely to be coreferent if they have more similar steps at the end. Because of that the aggregation takes into account the PTVs resulting from the coreference detection of the steps and the position of the steps in their respective paths. This rule is implemented by using the Sugeno integral for possibilistic truth values (8).

The aggregation operator for the comparison of two paths, A and B, is defined by the Sugeno integral for PTVs where:

- P is a set of propositions stating coreference of pairs of steps identified in the previous substep of the algorithm, i.e., 'mapping at the steps level',
- $\tilde{P}$ is the set of PTVs corresponding to the above mentioned pairs, representing the uncertainty about their truth values computed as discussed in the previous substep,
- the fuzzy measure γ^T is defined by Equation (11), where $Q \subseteq P$, $Q = \{p_1, \ldots, p_k\}$:

$$\gamma^T(Q) = \sum_{j=1}^{k} w_j, \text{ where } w_j = r_j / \sum_{j=1}^{k} r_j \qquad (11)$$

and w_j is the weight of the jth pair of steps and the ratio r_j equals

$$r_j = \frac{1}{2}\left(\left(\frac{pos(s_A^j)}{len(A)}\right)^{p(A,j)} + \left(\frac{pos(s_B^j)}{len(B)}\right)^{p(B,j)}\right) \qquad (12)$$

where step s_A^j is a part of path A, $pos(s)$ is the position of step s in its path, $len(A)$ is the length of path A (the number of steps in this path) and $p(A, j) = len(A) - pos(s_A^j)$. The exponentiation employed in (12) helps to spread weights. More precisely, thanks to it the distribution of weights follows the power law and weights for steps at the beginning of a path are lower than weights for steps at the end. This helps implementing the assumption that two paths are more likely to be coreferent if they have more similar steps at the end.

- the fuzzy measure γ^F is defined by

$$\gamma^F(Q) = \begin{cases} 1 & \text{if } Q = P \\ 0 & \text{otherwise} \end{cases} \qquad (13)$$

what is implied by condition (7) and the fact that γ^T is always greater than 0 because weights are always greater than 0.

Example. The aggregation at the steps level for the paths */cddb/disc/tracks/title* and */discs/disc/tracklist/title* (cf. Table II) goes as follows. First, the weights for the pairs of steps are computed, e.g., $w(cddb, discs) = 0.008$, $w(title, title) = 0.496$. These weights are then used to aggregate the PTVs from Table II. Table III shows the calculations for $Nec(\tilde{P}_{(i)^T} = T)$. The rows are sorted in decreasing order by ranking the PTVs expressing the coreference between matched pairs of steps. The first column shows matched steps. The second column shows the PTVs expressing their coreference, while column 4 gives the value for $Nec(\tilde{P}_{(i)^T} = T)$. For example, $Nec(\tilde{P}_{(i)^T} = T)$ in row 3 is computed by $Nec(\tilde{P}_{(i)^T} = T) = 1 - 0.16 = 0.84$. The third column shows the weights. The fuzzy measure γ^T of the subsequent sets $P_{(i)^T}$ is shown in column 5, e.g., for the steps pair $(tracks, tracklist)$ we obtain $\gamma^T(\tilde{P}_{(3)^T}) = 0.12 + 0.5 + 0.37 = 0.99$. The last column shows the minimum (denoted with $\wedge$ in (10)) of $\gamma^T(\tilde{P}_{(i)^T})$ and $Nec(\tilde{P}_{(i)^T} = T)$, e.g., for $(tracks, tracklist)$ we obtain 0.84. $Nec(\tilde{P}_{(i)^F} = F)$ is calculated analogously to $Nec(\tilde{P}_{(i)^T} = T)$. This time the rows are sorted in increasing order by ranking the PTVs as shown in Table IV. The last column equals 0 for all matched steps pairs. Hence, the coreference between the paths */cddb/disc/tracks/title* and */discs/disc/tracklist/title* is

TABLE III
$Nec(\tilde{P}_{(i)^T} = T)$ FOR PATHS */cddb/disc/tracks/title* AND */discs/disc/tracklist/title*.

Step	PTV	w_i	$Nec(\tilde{P}_{(i)^T} = T)$	$\gamma^T\left(\tilde{P}_{(i)^T}\right)$	$\wedge$
disc disc	T: 1.00 F: 0.00	0.12	1	0.12	0.12
title title	T: 1.00 F: 0.00	0.5	1	0.62	0.62
tracks tracklist	T: 1.00 F: 0.16	0.37	0.84	0.99	0.84
cddb discs	T: 0.00 F: 1.00	0.01	0	1	0

expressed by the PTV $\tilde{p}$ with $\mu_{\tilde{p}}(T) = 1 - 0 = 1$ and $\mu_{\tilde{p}}(F) = 1 - 0.84 = 0.16$ where 0 and 0.84 are respectively the maximum values ($\vee$) of the last columns of Tables IV and III (cf. Equation (8)).

TABLE IV
$Nec(\tilde{P}_{(i)^F} = F)$ FOR PATHS */cddb/disc/tracks/title* AND */discs/disc/tracklist/title*.

Step	PTV	w_i	$Nec(\tilde{P}_{(i)^F} = F)$	$\gamma^F(\tilde{P}_{(i)^F})$	$\wedge$
cddb discs	T: 0.00 F: 1.00	0.01	1	0	0
tracks tracklist	T: 1.00 F: 0.16	0.37	0	0	0
disc disc	T: 1.00 F: 0.00	0.12	0	0	0
title title	T: 1.00 F: 0.00	0.5	0	1	0

Step 3: Mapping algorithm at the paths level

One more time our algorithm establishes a mapping, but this time between paths. In this step the best 1:1 mapping between paths of two XML schemas is determined, based on the coreferent paths matrix of which an example is shown in Table I. The procedure is analogous to the *'Mapping at the steps level'* described above. Indeed, a coreferent paths matrix also consists of PTVs of which the largest in each row is selected hereby handling conflicts as described in Step 2, Substep 3. For example, the PTVs of the selected, matched paths are set in bold in Table I.

Step 4: Aggregation at the paths level

Finally, the coreference of the whole XML schemas can be computed. To this aim the aggregation of the PTVs expressing the paths coreference is done using a technique based on the Sugeno integral for PTVs [6], in a similar way as proposed for the *'Aggregation at the steps level'* described in Step 2, Substep 4. As argued earlier, this approach makes it possible to adequately cope with the different importance of aggregated elements, paths in this case. In [13] it is stated that not all attributes are equally important and not all of them have the same role in coreference discovery. For instance, it can be assumed that a path */cddb/disc/title* is more important than a path */cddb/disc/id* but this is, of course, context-dependent. Because of that, mapped elements are classified into subsets from the most to the least important. The classification can be done manually or be based on heuristics or knowledge stored in a knowledge base. So, PTVs expressing coreference of all matched paths can be aggregated using the Sugeno integral. The resulting aggregated PTV then expresses the (un)certainty about the coreference between the two input XML schemas, such as the left and right schemas represented in Figure 1.

IV. Evaluation and discussion

Datasets. To illustrate the proposed approach we consider two different real-world datasets, respectively containing information about 'compact discs' and 'university courses'. Compact disc data are represented by two schemas (also being used in the examples of the paper). The first schema is extracted from FreeDB and consists of 7 leaf elements. The second is extracted from Discogs and consists of 33 leaf elements of which 7 have been identified as being coreferent with FreeDB elements. University course data comprise three schemas and originate from the AnHai Doan repository[1]. They are derived from the websites of three universities: Reed College (Reed), University of Wisconsin-Milwaukee (UWM) and Washington State University (WSU). These schemas are considered in pairs: $(Reed, WSU)$ contains 12 coreferent elements, $(Reed, UWM)$ 10 and (WSU, UWM) 11. Real coreferent elements were manually identified based on the schemas and instance data.

Paths comparison. Table V presents the results of paths coreference detection using our method (referred to as *Paths Matcher*) and a two-level string comparison method proposed in [2]. The latter method is similar to the *Paths Matcher*, but does not take into account the positions of path elements. Table V presents the PTVs denoting (un)certainty of coreference for selected pairs of paths, calculated using both methods. Columns correspond to particular methods and rows contain selected paths from the CD and university courses datasets. Path pairs from the CD datasets have equal number of steps and different steps at the beginning, but are more similar at the end. As Paths Matcher takes into account this feature of coreferent paths, it gives much better results for these pairs than the other method. However, when paths are similar at the beginning, the results obtained using Paths Matcher are only slightly better. Thus, our method properly implements the assumption that not all steps are equally important and differences at the beginning of paths do not exclude paths coreference.

Paths Matcher properly deals with the differences in the schema structures (e.g., the fourth row of Table V), abbrevations (e.g., *tracklist* and *tracks* in the second row) and misspellings. This is confirmed by the larger PTVs returned by our method for truly coreferent elements (e.g., */cddb/disc/year* and */discs/disc/year* in the first row of Table V).

Recall and precision. Precision is one of the important measures of quality for classification results. In our case it can be defined as the fraction of really coreferent objects among all objects classified by a given algorithm as being coreferent (*true*

[1]Doan, AnHai, http://www.cs.washington.edu/research/xmldatasets

TABLE V
COMPARISON OF TWO ALGORITHMS FOR DETECTING PATHS COREFERENCE.

Paths	Paths Matcher	Two-Level Matcher
/cddb/disc/year	**T: 1.00**	T: 1.00
/discs/disc/year	**F: 0.06**	F: 0.25
/cddb/disc/tracks/title	**T: 1.00**	T: 1.00
/discs/disc/tracklist/title	**F: 0.17**	F: 0.42
/root/course/instructor	T: 1.00	T: 1.00
/root/courseListing/restrictions	F: 0.56	F: 0.12
/root/course/days	**T: 1.00**	T: 1.00
/root/courseListing/sectionListing/days	**F: 0.33**	F: 0.67

positives). High precision means that the method returns more relevant (coreferent) than irrelevant results. Recall is another important quality measure which in our case can be defined as the fraction of true positive objects among all coreferent objects present in a test dataset. High recall means that a method is capable to discover most of the actually relevant objects. Our algorithm employs two thresholds to decide on paths coreference, which are set to 0.55 for $\mu_{\tilde{p}}(T)$ and to 0.5 for $\mu_{\tilde{p}}(F)$. If $\mu_{\tilde{p}}(F)$ is lower than the threshold then the coreference is declared. If $\mu_{\tilde{p}}(T)$ is lower than the threshold then the lack of coreference is declared. Finally, if both of the thresholds are exceeded then the coreference status is declared as *unknown*.

Table VI presents the recall and precision for the datasets used in our experiments and calculated for the results obtained using our algorithm (*Paths Matcher*), the two-level string comparison[2] (*Two-Level Matcher*) and a Levenshtein distance [14] (*Levenshtein*) based approach which employs a predefined threshold value 0.5 on the normalized number of edit operations needed, to decide if the coreference occurs. *Paths matcher* returns better results than other methods for each of different pairs of datasets. Our method gives the best result compared to the other methods for the CD datasets, because paths in these datasets have different beginnings and our method copes with that type of differences properly. Moreover, schemas of the Reed and WSU datasets are similar so all methods return fairly good results, as shown in row two of Table VI. The last two rows contain results of comparing schemas that most differ in structure and names of their elements. This explains the low results obtained for them in our evaluation.

TABLE VI
RECALL AND PRECISION.

Datasets	Paths Matcher		Two Level		Levenshtein	
	Precis.	Recall	Precis.	Recall	Precis.	Recall
FreeDB Discogs	**0.8**	**0.57**	0.14	0.14	0.5	0.14
Reed WSU	**1.0**	0.75	0.75	0.75	0.83	**0.83**
Reed UWM	**0.5**	**0.3**	0.13	0.1	0.25	0.1
WSU UWM	**0.67**	**0.36**	0.25	0.18	0.38	0.27

V. CONCLUSION

We proposed a method to find coreferent XML elements based on coreferent paths what may help to further decide on the coreference of XML schemas. Paths are one of the most crucial metadata in XML files. Detection of coreferent paths requires recognizing coreferent steps from which paths are constructed. We treat coreference as a binary notion, i.e., two paths are actually coreferent or not. However, we assume that the results of the coreference detection may be uncertain what is represented by employing possibilistic truth values. Coreference is considered here in a hierarchical way. On the basic level, the coreference of steps (parts of XML paths) is determined. Then, the information on steps coreference, with an explicit representation of its related uncertainty using PTVs, is properly aggregated to obtain information on paths coreference which is on its turn further aggregated to finally decide on the coreference of whole XML schemas. We apply a Sugeno integral for PTVs to aggregate information on the coreference of subsequent levels of this hierarchy. This allows us to explicitly cope with the position of the elements in a path and the relative importance of paths within their schema.

ACKNOWLEDGMENT

This contribution is supported by the Foundation for Polish Science under International PhD Projects in Intelligent Computing. Project financed from The European Union within the Innovative Economy Operational Programme 2007-2013 and European Regional Development Fund.

REFERENCES

[1] D. C. Fallside, XML Schema Part 0: Primer, 2001.
[2] A. Bronselaer and G. De Tré, "A possibilistic approach to string comparison," IEEE TRANS. ON FUZZY SYST., vol. 17, no. 1, pp. 208–223, 2009.
[3] L. A. Zadeh, "Fuzzy sets as a basis for a theory of possibility," Fuzzy Sets Syst., vol. 100, pp. 9–34, Apr. 1999.
[4] H. Prade, "Possibility sets, fuzzy sets and their relation to Lukasiewicz logic," in Proc 12th Int Symp on Multiple-Valued Logic, 1982, pp. 223–227.
[5] E. P. Klement, R. Mesiar, and E. Pap, Triangular Norms, 1st ed. Springer, 2000.
[6] A. Bronselaer, A. Hallez, and G. D. Tré, "Extensions of fuzzy measures and Sugeno integral for possibilistic truth values," Int. J. Intell. Syst., vol. 24, no. 2, pp. 97–117, Feb. 2009.
[7] M. Sugeno, "Theory of fuzzy integrals and its applications," Ph.D. dissertation, Tokyo, Japan, 1974.
[8] M. Szymczak, A. Bronselaer, S. Zadrozny, and G. De Tré, "Dynamical construction of binary relations in coreference detection," in NAFIPS 2012. IEEE, 2012, pp. 100–106.
[9] A. Bronselaer, A. Hallez, and G. De Tré, "Evaluation in the possibilistic framework for object matching," in IPMU 2008, 2008.
[10] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate record detection: A survey," IEEE Trans. Knowl. Data Eng., vol. 19, no. 1, pp. 1–16, 2007.
[11] P. Calado, M. Herschel, and L. Leitão, "An overview of xml duplicate detection algorithms," in Soft Comput. in XML Data Manag., 2010, pp. 193–224.
[12] S. Zadrozny, J. Kacprzyk, and G. Sobota, "Avoiding duplicate records in a database using a linguistic quantifier based aggregation - a practical approach," in FUZZ-IEEE. IEEE, 2008, pp. 2194–2201.
[13] P. Zadeh and M. Reformat, "Fuzzy semantic similarity in linked data using the OWA operator," in NAFIPS 2012, Aug. 2012, pp. 1–6.
[14] V. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," Soviet Physics Doklady, vol. 10, p. 707, 1966.